1. Design a Currency Converter Application. The program converts US Dollars to Brazilian Real, Canadian Dollars, European Union Euros and Japanese Yen.

Create the GUI as shown below. Perform object oriented system testing considering the use cases mentioned.



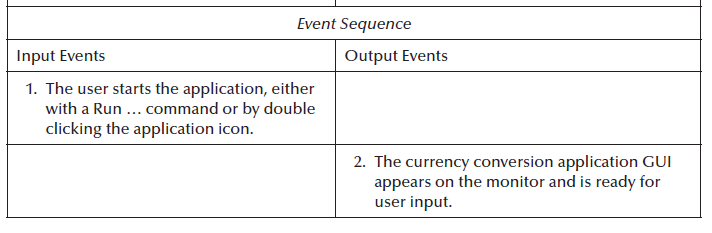
Currency converter User interface

Use cases:

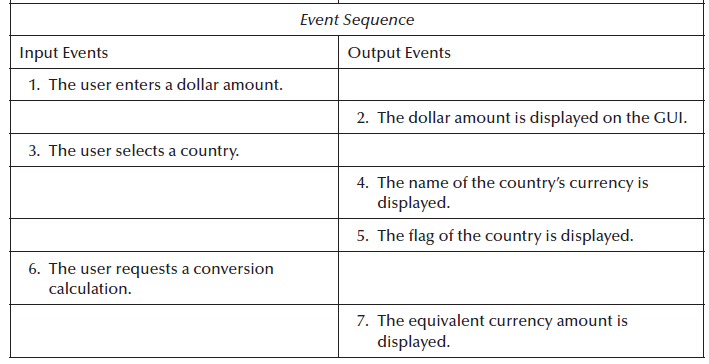
|  |  |  |
| --- | --- | --- |
| Sl. No. | Name | Description |
| 1 | Start application | The user starts the currency conversion application in Windows |
| 2 | End application | The user ends the currency conversion application in Windows |
| 3 | Convert dollars | The user inputs a US dollar amount and selects a country; the application computes and displays the equivalent in the currency of the selected country |
| 4 | Revise inputs | The user resets inputs to begin a new transaction |
| 5 | Abnormal case: no country selected | User enters a dollar amount and clicks on the Compute button without selecting a country |
| 6 | Abnormal case: no dollar amount entered | User selects a country and clicks on the Compute button without entering a dollar amount |
| 7 | Abnormal case: no dollar amount entered and no country selected | User clicks on the Compute button without entering a dollar amount and without selecting a country |

Some of the events are mentioned below:

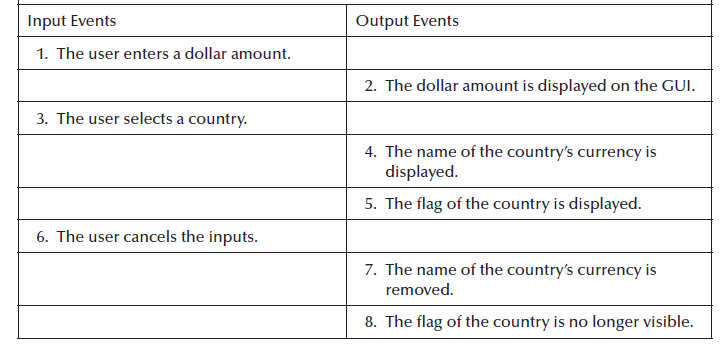
Sequence of events for use case: Start application



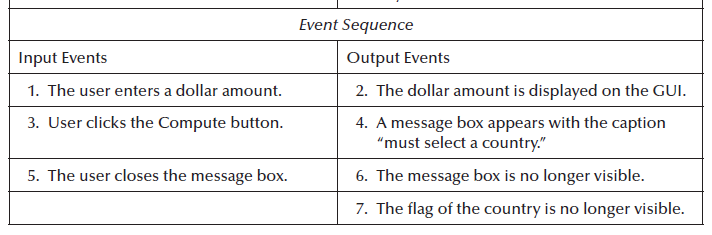
Sequence of events for use case: Convert Dollars



Sequence of events for use case: Revise inputs



Sequence of events for use case: Abnormal case: no country selected



Code:

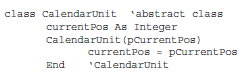
import javax.swing.\*;  
import java.awt.\*;  
import javax.swing.JOptionPane;  
import java.awt.event.\*;  
  
public class stlab {  
  
 public static void converter()  
 {  
 JFrame f = new JFrame("CONVERTER");  
   
 JLabel l1, l2;  
 JTextField t1, t2;  
   
 JButton b1, b2, b3;  
   
 l1 = new JLabel("Dollars:");  
 l1.setBounds(20, 40, 60, 60);  
 l2 = new JLabel("Equivalent in:");  
 l2.setBounds(170, 40, 60, 60);  
   
 t1 = new JTextField();  
 t1.setBounds(80, 40, 70, 40);  
 t2 = new JTextField();  
 t2.setBounds(240, 40, 70, 40);  
   
 b1 = new JButton("Compute");  
 b1.setBounds(50, 150, 100, 30);  
   
 Choice c=new Choice();   
 c.setBounds(150,110,80,100);   
 c.add("Brazil");   
 c.add("Canada");   
 c.add("European Community");   
 c.add("Japan");  
 c.add(" ");  
   
 b2 = new JButton("Clear");  
 b2.setBounds(240, 150, 70, 30);  
   
 b3 = new JButton("Quit");  
 b3.setBounds(170, 150, 60, 30);  
   
 b1.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e)  
 {  
 double d1 = 0;  
 try {  
 double d = Double.valueOf(t1.getText());  
 if(c.getItem(c.getSelectedIndex()).equals("Brazil"))  
 {   
 d1 = (d \* 5.23);  
 }  
 else if(c.getItem(c.getSelectedIndex()).equals("Canada"))  
 {  
 d1 = (d \* 1.21);  
 }  
 else if(c.getItem(c.getSelectedIndex()).equals("European Community"))  
 {  
 d1 = (d \* 0.82);   
 }  
 else if(c.getItem(c.getSelectedIndex()).equals("Japan"))  
 {  
 d1 = (d \* 108.83);   
 }  
 else if(c.getItem(c.getSelectedIndex()).equals(" "))  
 {  
 JOptionPane.showMessageDialog(f,"Please add the country.","Warning",JOptionPane.WARNING\_MESSAGE);  
 }  
 else  
 {  
 d1 = 0;  
 }  
 }  
 catch(NumberFormatException a) {  
 JOptionPane.showMessageDialog(f,"Invalid Entry. Please check the amount or country","Warning",JOptionPane.WARNING\_MESSAGE);  
 }  
   
 String str1 = String.valueOf(d1);  
 t2.setText(str1);  
 }  
 });  
   
 b2.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e)  
 {  
 t1.setText(" ");  
 t2.setText(" ");  
 }  
 });  
   
 b3.addActionListener(new ActionListener() {  
 public void actionPerformed(ActionEvent e)  
 {  
 f.dispose();  
 }  
 });  
   
 f.addWindowListener(new WindowAdapter() {  
 public void windowClosing(WindowEvent e)  
 {  
 System.exit(0);  
 }  
 });  
   
 f.add(l1);  
 f.add(t1);  
 f.add(l2);  
 f.add(t2);  
 f.add(b1);  
 f.add(b2);  
 f.add(b3);  
 f.add(c);  
   
 f.setLayout(null);  
 f.setSize(800, 800);  
 f.setVisible(true);  
 }  
   
  
 public static void main(String args[])  
 {  
 converter();  
 }  
}

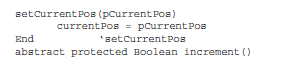
2. The Calendar application has classes: CalendarUnit, testIT, Date, Day, Month, Year. Perform object oriented integration testing.

Structure of classes are given below:

**Class: CalendarUnit**

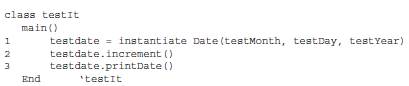
**Responsibility: Provides an operation to set its value in inherited classes and provides a Boolean operation that tells whether an attribute in an inherited class can be incremented.**





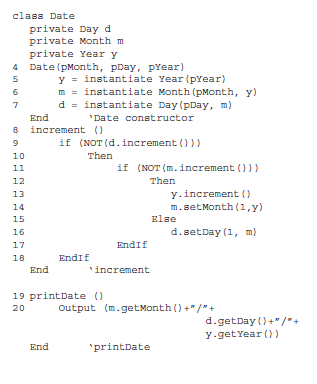
**Class: testIt**

**Responsibility: Serves as a test driver by creating a test date object, then requesting the object to increment itself, and finally, to print its new value.**



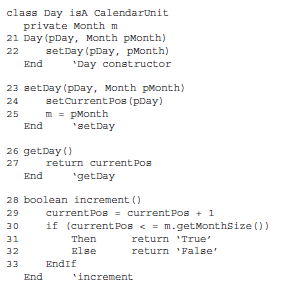
**Class: Date**

**Responsibility: A Date object is composed of day, month, and year objects. A Date object increments itself using the inherited Boolean increment methods in Day and Month objects. If the day and month objects cannot be incremented (e.g., last day of the month or year), Date’s increment method resets day and month as needed. In the case of December 31, it also increments the year. The printDate operation uses the get() methods in Day, Month, and Year objects and prints a date value in mm/dd/yyyy format.**



**Class: Day**

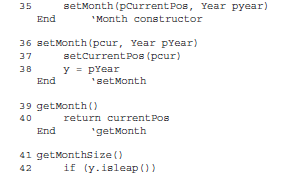
**Responsibility: A Day object has a private month attribute that the increment method uses to see if a day value can be incremented or reset to 1. Day objects also provide get() and set() methods.**

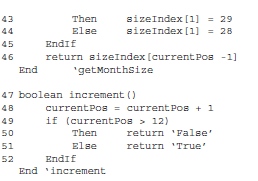


**Class: Month**

**Responsibility: Month objects have a value attribute that is used as a subscript to an array of values of last month days (e.g., the last day of January is 31, the last day of February is 28, and so on). Month objects provide get() and set() services, and the inherited Boolean increment method. The possibility of February 29 is determined with the isleap message to a Year object.**

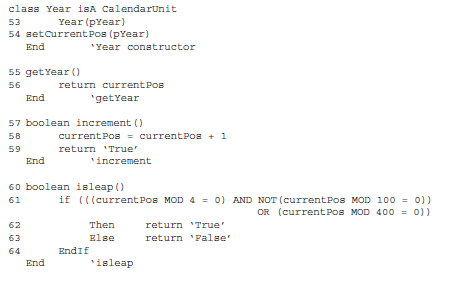






**Class: Year**

**Responsibility: In addition to the usual get() and set() methods, a Year object increments itself when the test date is December 31 of any year. Year objects provide a Boolean service that tells whether the current value corresponds to a leap year.**

****

Integrate the classes based on collaboration diagram given below:

